Maratona de Programação da SBC 2025

Sub-Regional Brasil do ICPC

13 de setembro de 2025

Sessão de Aquecimento

Limites de tempo

Antes da competição, os juízes terão resolvido todos os problemas em linguagens de pelo menos dois dos três grupos distintos de linguagens (C/C++, Java/Kotlin e Python3). Os limites de tempo de cada problema são calculados com base no tempo de execução dessas soluções.
Os tempos são dados em segundos:

Problema	Tempo Limite por Caso de Teste*
A	0.5
B	1.0
C	1.5
D	0.5

`* Note que os limites de tempo não variam conforme a linguagem de programação`

Limites de memória

```
C, C++20, Python: 1GB
Java, Kotlin: 1GB + 100MB stack
```

Outros limites

```
Tamanho do arquivo fonte: 100KB
```

Comandos de compilação

```
C: gcc -x c -g -O2 -std=gnu11 -static -lm
C++20: g++ -x c++ -g -O2 -std=gnu++20 -static
Java: javac
Python: pypy3 -m py_compile
Kotlin: kotlinc -J-Xms1024m -J-Xmx1024m -J-Xss100m -include-runtime
```

C/C++

- Seu programa deve retornar zero, executando, como último comando, `return 0` ou `exit(0)`.
- É sabido que em alguns casos de problemas com entradas muito grandes, os objetos `cin` podem ser lento, por conta da sincronização de buffers da biblioteca `stdio`. Caso deseje-se utilizar `cin` e `cout`, um jeito de se contornar este problema é executando-se o comando `ios_base::sync_with_stdio(0)`, no início de sua função `main`. Note que, neste caso, o uso de `scanf` e `printf` no mesmo programa é contra-indicado, uma vez que, com buffers separados, comportamentos inadequados podem ocorrer.

Java

- Não declare '`package`' no seu programa Java.
- Note que a convenção para o nome do arquivo fonte deve ser obedecida, o que significa que o nome de sua classe pública deve ser uma letra maiúscula (A, B, C, etc).
- Comando para executar uma solução Java:
  `java -Xms1024m -Xmx1024m -Xss100m {codigo_do_problema}`

Kotlin

- Não declare '`package`' no seu programa Kotlin.
- Note que a convenção para o nome do arquivo de solução deve ser obedecida, o que significa que seu arquivo-fonte deve ser nomeado (A.kt, B.kt, C.kt, etc).
- Comando para executar uma solução Kotlin:
  `java -Xms1024m -Xmx1024m -Xss100m {codigo_do_problema}Kt`

Python

- Note que apenas Python 3 é suportado.
- As submissões em Python terão sua sintaxe verificada; programas que não passarem nessa verificação receberão o veredito "Compilation Error".
- Note que, este ano, as submissões em Python serão executadas com o interpretador **PyPy**, em vez do CPython tradicional. Isso geralmente resulta em melhor desempenho, mas esteja atento a eventuais diferenças sutis de comportamento.
- Comando para executar uma solução Python: `pypy3 {fonte}.py`

Instruções para uso do Sistema de Submissão Boca

Submissão de soluções

Para envia uma solução para um problema, você deve usar a interface web do Boca:

- Abra o navegador.
- Faça o login como um time (use o nome de usuário e senha fornecidos).
- Acesse a aba `Runs`. Escolha o problema apropriado, a linguagem utilizada e envie o arquivo fonte.

Resultado da submissão

Para ver o resultado de uma submissão, você deve usar a interface web do Boca:

- Abra o navegador.
- Faça o login como um time (use o nome de usuário e senha fornecidos).
- Acesse a aba `Runs`.

Os vereditos que você pode receber dos juízes são

- 1 - YES
- 2 - NO - Compilation error
- 3 - NO - Runtime error
- 4 - NO - Time limit exceeded
- 5 - NO - Wrong answer
- 6 - NO - Contact staff

Os significados de 1, 2, 3, 4 e 5 são auto-explicativos.

- Sobre 1 e 5:
  - se a saída da solução do time for exatamente igual à saída dos juízes, ou, em problemas nos quais esteja especificado que múltiplas soluções são aceitas e a sua solução atenda às definições descritas no enunciado, a resposta será "YES";
  - caso contrário, o veredito é "Wrong Answer".
- Sobre 6: usado em circunstâncias inesperadas. Neste caso, utilize o menu "Clarifications" e fornaça o número da "run" para maiores esclarecimentos.

Seu programa pode ser executado em múltiplos arquivos de entrada. Note que isso significa que, se o seu programa tiver mais de um erro (por exemplo, "Time Limit Exceeded" e "Wrong Answer"), você poderá receber qualquer um desses erros como veredito.

A formatação da saída deve seguir o exemplo de saída fornecido no enunciado do problema, embora espaços em branco extras, dentro do razoável, sejam aceitáveis. Por exemplo, se você imprimir milhares de espaços em branco dentro do limite de tempo e saída do problema, isso será julgado como "Wrong Answer"; no entanto, um espaço extra no final de uma linha ou uma linha em branco adicional são aceitáveis.

Se o seu programa gerar saída em excesso, sua submissão receberá um veredito de "Runtime Error". Note que qualquer conteúdo escrito na saída de erro padrão (*stderr*) também conta para esse limite. Portanto, um excesso de mensagens de depuração pode causar erros de execução. Se você encontrar um erro de execução inesperado, considere reduzir os logs em *stderr*.
Isso também se aplica a problemas interativos, abrangendo tanto a saída que sua solução envia ao juiz interativo quanto os logs em *stderr*.

Note que, no caso de submissões em Java ou Kotlin, é necessário seguir a convenção para o nome do arquivo-fonte (e da classe). Além disso, não deve haver declaração de `package`. Caso essas regras não sejam respeitadas, a submissão poderá receber um veredito de "Compilation Error" ou até mesmo de "Runtime Error".

Penalidades

Cada submissão para um problema que receba um veredito "`NO`" antes do primeiro veredito "`YES`" para o mesmo problema acarretará uma penalidade de tempo de 20 minutos, adicionada ao tempo total do time. Não há penalidade de tempo para problemas não resolvidos.

As exceções a essa regra são "`NO - Compilation error`" e "`NO - Contact staff`", que não possuem penalidade de tempo associada.

Problemas interativos

É possível que o conjunto de problemas contenha zero ou mais problemas interativos.

Em muitos aspectos, problemas interativos são como qualquer outro: seu programa deve ler da entrada padrão e escrever na saída padrão. A diferença é que, nesse caso, a entrada e a saída do seu programa estão conectadas a outro programa (o juiz), com o qual ele deve se comunicar de forma contínua, enviando dados e recebendo respostas.

Para isso, é crucial que o seu programa "force" a saída de dados imediatamente após cada envio (use operações como `fflush(stdout)` em C, `cout.flush()` em C++, `System.out.flush()` em Java/Kotlin ou `sys.stdout.flush()` em Python). Caso não faça isso, a comunicação pode não funcionar corretamente, resultando em vereditos como "Wrong Answer" ou "Time Limit Exceeded".

Além disso, tenha atenção aos seguintes pontos:

- O enunciado do problema define o **protocolo de interação**: quem começa, qual o formato de cada mensagem, e como o diálogo deve prosseguir. Seu programa deve **seguir esse protocolo exatamente**, respeitando espaços, quebras de linha e formatos.
- Se seu programa deixar de ler a resposta do juiz, ou tentar ler quando não há nada disponível, pode ocorrer um travamento (*deadlock*) e o veredito será incorreto.
- Saídas mal formatadas ou em ordem incorreta podem levar o juiz a encerrar a execução com veredito de "Wrong Answer".
- Interações devem ser realizadas de forma eficiente. Se o programa gastar muito tempo sem responder, pode receber veredito de "Time Limit Exceeded", mesmo que o algoritmo esteja correto.
- O juiz pode se comportar de maneira adversarial, adaptando as entradas às suas respostas para tentar expor erros no algoritmo.

Esclarecimentos

Para solicitar esclarecimentos sobre o enunciado de um problema, você deve usar a interface web do Boca:

- Abra o navegador.
- Faça o login como um time (use o nome de usuário e senha fornecidos).
- Acesse a aba `Clarifications`. Escolha o problema apropriado e digite sua questão.

Placar

Para ver o placar local você deve usar a interface web do Boca:

- Abra o navegador.
- Faça o login como um time (use o nome de usuário e senha fornecidos).
- Acesse a aba `Score`. Você terá acesso ao placar local.